Unconstrained Real-time Markerless Hand Tracking for Humanoid Interaction

Thomas Gumpp[‡], Pedram Azad[*], Kai Welke[‡*], Erhan Oztop[†‡], Rüdiger Dillmann[*], Gordon Cheng[‡†]
[‡]Department of Humanoid Robotics and Computational Neuroscience
ATR Computational Neuroscience Laboratories
[†]JST-ICORP, Computational Brain Project,
4-1-8 Honcho, Kawaguchi, Saitama, Japan.
Email: {gumpp, welke, erhan, gordon}@atr.jp
[*]Institute for Computer Science and Engineering, University of Karlsruhe, Germany
Email: {azad, dillmann}@ira.uka.de

***Abstract*— Markerless hand tracking of humans can be applied to a broad range of applications, in robotics, animation and natural human-computer interaction. Traditional motion capture and tracking methods involve the usage of devices such as a data glove, or marker points that are fixed and calibrated on the object to perform tracking. Markerless tracking is free from such needs, and therefore allows for more freedom in movement and spontaneous interaction. In this paper, we analyze how a hand tracking system, which reliably tracks arbitrary hand movements can be implemented. We explored a model based approach that uses particle filters for tracking. In this study we also determine the degree to which the inherent parallel properties of particle filter can be exploited to achieve the goal of real-time tracking. We present the effectiveness of the tracking system via the real-time control of a 20 degrees of freedom dextrous robotic hand.**

I. Introduction

For any robot that is designed to interact in a natural manner with humans, an unconstrained hand-tracker is necessary. Abilities that a humanoid can exhibit using a markerless hand tracker range from skill learning, imitation to cooperative human interaction, such as handling and exchanging objects with humans via hand-eye coordination, and enable comprehension of gestures. Humans can also use this system to direct the attention of a robot by pointing. A hand tracker system presented in this article addresses the necessities of these tasks.

II. State of the art

Marker-based and glove-based hand-trackers have been thoroughly investigated (for a survey, see [1]). These techniques usually lead to good tracking results. For our application, the robot should be able to track any random hand in a natural environment, therefore it cannot rely on any preparations such as fixed markers. For this reason, we opt for a markerless solution.

Markerless hand tracking is usually based on the interpretation of a video input stream. The grabbed frames are searched for cues. Common cues are the outline of the hand, which can be found by applying an edge-filter on the input frame. Another cue is the visible area of a hand, which can be extracted by filtering skin color. The tracker's estimated hand configuration is compared to cues in the input image to get a probability for a defined gesture. Either a posture database or a projected model of a 3D hand model is used to compare against the cues.

Recognition of predefined hand postures is solved by systems that usually use a posture database. One example is the appearance based gesture recognition [2] using Hidden Markov Models to detect gestures in a sign language database (see [3] for a review of similar systems).

More closely in line with our own objectives are multi-degrees of freedom hand-trackers, which have recently been reviewed thoroughly by Erol in [4]. The review presents recent developments in this field distinguished by three categories: Single Frame Estimators, Single- and Multiple Hypotheses Estimators.

A. Single Frame Estimators

The first class contains single frame pose estimators that do not rely on prior knowledge from former frames. One possible solution is global search over a set of templates labeled with the kinematic parameters. In [5], these templates are generated using a 3D model, and stored in a tree structure. To detect a pose, the tree is traversed without prior knowledge. Being independent from former frames, these systems do not have to recover from tracking errors in a former frame, resulting in a robust behavior. Building up a template database limits the system's capability of expression to a discrete subset of postures, which contradicts our goal of unconstrained tracking. Global search is computationally very expensive. Using a continuous joint space to track is not feasible for a real-time system. Tracking over multiple frames could result in a system that uses less resources, as only local search is performed in configuration space.

B. Single Hypothesis Estimators

The second category contains hand trackers that use a single hypothesis carried over from former estimations as prior knowledge. The most common approach for fitting a model to cues extracted from the image is to use standard optimization techniques. In [6], the error based on joint links and finger tips was minimized using Newton's method. Silhouette-based

error measures were minimized using Nelder Mead Simplex method in [7].

In [8], a divide and conquer approach was proposed. First, the overall motion of the hand was estimated, followed by the estimation of the joint angles. This procedure was iteratively applied until convergence. Kalman Filtering has also been used to solve the single hypothesis tracking problem. In [9], the Unscented Kalman Filter (UKF) was used for hand tracking. As these systems rely on gradient search, there is always the danger of reaching only a local minimum without finding the global minima over the tracking error landscape.

C. Multiple Hypotheses Estimators

Opposed to single hypotheses trackers, multiple hypotheses trackers try to keep multiple pose estimations while tracking a sequence. This increases the chance of reaching the global minima. This idea is best implemented by the Bayesian filtering framework that keeps a probability distribution of the states up to the current frame. One prominent technique that implements a recursive Bayesian filter is Particle Filter.

Particle filtering is also known as the Condensation algorithm introduced by Michael Isard and Andrew Blake [10]. The Condensation algorithm was designed to track curves in clutter. Kalman filtering is not suitable for this task, since it is based on Gaussian densities which are not capable of simultaneously representing alternative hypotheses.

A particle is a pair $(s_n\ ,\ \pi_n)$, where s_n is the configuration vector of particle n, and π_n is the probability of this configuration. For hand trackers, each particle consists of a configuration vector and its probability, and each value in the configuration vector represents an angle or a translation of the kinematic hand model. The vector's dimension is therefore equal to the number of DOF of the hand kinematic. π_n is calculated by rating the state of s_n. This can be done by evaluating different cues like visible areas or edges in video frames. The probability distribution is modeled by a set $S = (s_1, \pi_1), \ldots, (s_N, \pi_N)$ of particles.

The main steps for each iteration of the Particle Filter are sampling the set of particles S_t from the last iteration set S_{t-1}, rate each s_n to get its probability π_n and compute a weighted mean s_{mean}. s_{mean} represents the currently assumed estimation for the hand position and posture (see [11] for an application to finger pose tracking).

Although these methods are computationally more expensive than single hypotheses trackers, we decided to use a particle filter for tracking. The computational costs can be accounted for with balanced parallel execution, as the additional cost comes in the form of independent probability calculations.

In Section III, we discuss requirements for implementing Hand-Tracking on a Humanoid. In Section IV, we will present our particle filter based hand tracker. We propose a novel way to deal with self occlusion of an articulated hand model using a *Z-Buffer* method. We also propose a search space decomposition method to improve tracking performance. In Section V, we will show tracking results, and conclude the paper in Section VI.

III. Implementing Humanoid Hand-Tracking

A. Unconstrained Tracking

Gesture recognition systems that learn views of gestures are an example of a specialised hand tracker.

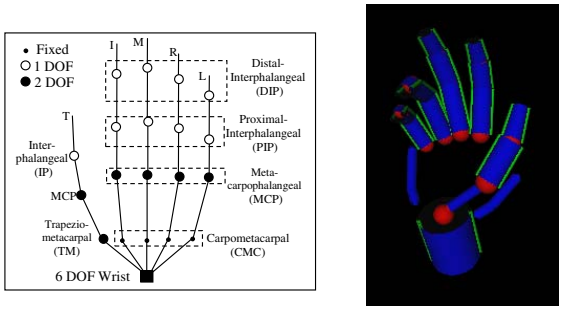
Such a system, like [2], can perform well in its domain, but is not designed to track arbitrary hand configurations, which humanoid robots necessitate since interacting with humans must take place without artificially introduced limitations. Our approach is extracting an unbiased representation of the tracked hand configurations, that enables the behavior module of our humanoid to react to and to interpret hand movements. It can be used as a basis for various tasks with different requirements such as imitation, direction of attention and estimation of intention, hand eye coordination, and gesture recognition.

To achieve this goal, we have chosen a model-based approach using a model similar to [12]. By using a 3D model as a means of configuration representation, we are not limited to a fixed set of hand postures. Shimada [13] uses a 3D model to build a labeled database of hand posture templates, which are then used for tracking. The result of this approach is primarily a hand pose, with discrete kinematic parameters attached. In contrast, we directly project our 3D model into the scene, so we know the exact kinematic parameters in a continuous joint angle range. Using a database is generally computationally cheaper, however it requires substantial memory and search time in return for less computation. There are several advantages in using a model directly. We can alter its structural parameters dynamically or completely exchange the model to adjust to different hands. We are also free to change projection details on-the-fly, especially important for humanoid robotic heads (see [14]), where foveal and wide angle camera systems with different projection characteristics are used. In contrast, a template based system would need a seperate database for each camera. Additionally, the degrees of freedom of the model can be adjusted for different tasks. Since all postures are generated on the fly, this system does not suffer from the curse of dimensionality that leads to exponential memory requirements for a posture database.

B. Real-time Requirements

One of the conclusions of [4] was the scarcity of real-time 3D hand-trackers. We need to create a real-time system as we have to interact with our environment and with people, so it is not acceptable to introduce long delays in the tracking process. Our goal is to build a real-time tracking system, to establish a natural means of communication for robots and humans.

With this requirement in mind, the particle filter system seems promising. It can scale very well as it is easily parallelised in a way that minimizes necessary communication between distributed system parts. Exploiting this property, we were able to implement a distributed particle filter on the base of the Distributed Vision Cluster, proposed in [15]. Like in [13] the computationally most expensive task is to calculate the probability π_n for each particle. We chose a



(a)

(b)

Fig. 2. (a) Structure of our Hand Model(Figure taken from [4]), and (b) 3D Hand Model

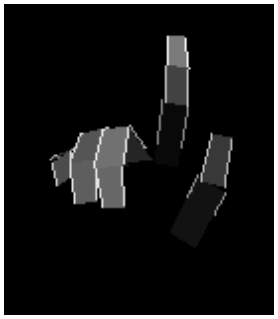


Fig. 3. Z-buffer. The depth information is displayed via the intensity of edges and areas.

Fig. 4. 3D Model Projection. The left image shows the extracted skin color image Φ^a, and the right image shows the extracted edges in Φ^e.

similar configuration as [13], with one computer used to manage tracking and for grabbing and preprocessing images, and several cluster nodes to compute probability functions (see, Figure 5).

One application that needs real-time is imitation. We used the Gifu Robotic Hand III (Dainichi Co. Ltd., Japan) to imitate hand postures. To achieve this goal, we convert the estimated angles of our tracker system to the joint angles of the Gifu Hand. See Figure 1 for results.

IV. The Markerless Tracker System

A. Image Acquisition and Preprocessing

In our system, we use a calibrated stereo camera system with small intra-ocular distance, as it can be found in a humanoid head. We have chosen stereo cameras, because two viewpoints of the hand lead to a better positioning in 3D space, as an error in depth estimation might be negligible in one view, but visible in the second view.

After grabbing the stereo image, we use skin color detection in HSV space to generate a binary map for skin-color area in the image, which we also use to rate the skin area cue. Then we use this mask to cut all uninteresting parts of the image, and apply an edge filter in the resulting area of the original image to obtain a hand edge image.

B. 3D Hand Model

Although the human hand has 22 DOF, we have chosen to limit the degree of freedom in our kinematic hand model to one degree of freedom per finger for flexing, to decrease complexity. As observed by [4], we set $\theta_{DIP} = \frac{2}{3}\theta_{PIP}$ for each finger. Additionally, our tracker assumes $\theta_{PIP} = \theta_{MCP}$, to decrease the dimensionality of search-space. The overall model consists of six degrees of freedom for hand orientation and position, and two degrees of freedom for each finger. In the current version of our system, the thumb is fixed in a standard position. So the configuration vector s_n of each particle has the dimension 14.

We use a 3D hand model (see, Figure 2) which only consists of conic sections. This model can be projected very fast into frames for cue detection, but is still precise enough to track hand and finger motion.

C. Tackling Self Occlusion

Deutscher proposed a method to rate image cues [16]. He directly rated each projected body part in the images. In order to avoid searching for self occluded parts of the hand model projection, we modified his method. We are proposing a z-buffer approach to only search for visible, not occluded cues.

The idea of z-buffering is to draw occluded objects so that objects that are further away get drawn earlier and are overwritten by the nearer occluding objects. To be able to use z-buffering to correctly project our 3D hand model, we need an ordered list of finger parts sorted by their distance to the projection plane. Because this projection plane is not the same in a stereo camera system, we need to create a separate z-buffer for each camera. To project the 3D hand model element, we use the inverse of the calibration matrix of each camera. As an approximation, we compare the centre of gravity of each finger, transformed into the coordinate system of the viewpoint. We then sort the finger parts starting from the biggest z coordinate, which is now perpendicular to the projection plane. After generating the list, we draw each element of the fingers according to the list's order in a buffer thereby coding the finger part, and whether it represents an edge or an area cue. By doing this, we are overwriting self-occluded cue indexes with those from overlapping fingers, as both will write to the same position in the buffer.

We then take a buffer as illustrated in Figure 3, where for each pixel we know whether we expect an edge, area, or nothing. These projections are finally used for comparison with the preprocessed images.

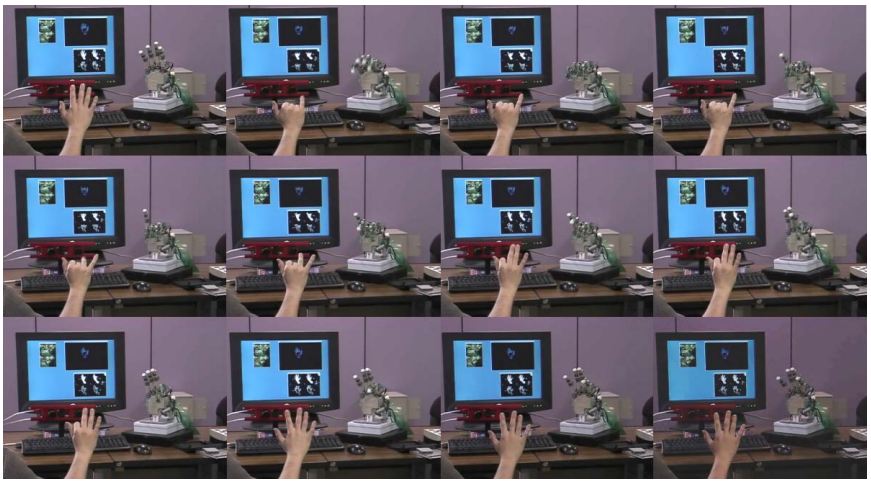


Fig. 1. Imitation sequence using the Gifu Robotic Hand

D. Rating Configuration Probability

We rate each particle's configuration based on edge and area cues found in the preprocessed frames. To compute edge probability π_n^e and area probability π_n^a of each particle n, we use

$$\mu_n^a = \sum \Phi^a[i], \forall i \in \Pi_n^a \tag{1}$$

$$\mu_n^m = |\Pi_n^a| - \mu_n^a \tag{2}$$

$$\pi_n^a = \frac{max((\mu_n^a - \mu_n^m), 0)}{|\Phi^a|} \tag{3}$$

$$\pi_n^e = \frac{\sum \Phi^e[i]^2}{|\Pi_n^e| * MaxIntensity^2} \forall i \in \Pi_n^e \tag{4}$$

where Φ^a is the the binary skin mask, μ_n^a and μ_n^m contain actually found and missing skin area pixels for the particle n, Φ^e is the mask-cut gradient image and $\Pi_n^{a/e}$ contains all area/edge in the z-buffer for particle n.

We also detect collisions between fingers by computing the shortest distance between two finger parts. If this distance is shorter than the sum of the radii, we assume a collision occurred and punish this state in the probability function via π_n^{coll}. Each particle's probability π_n is then computed using

$$\pi_n = e^{-\lambda(1-(\omega[0]\pi_n^e + \omega[1]\pi_n^a + \omega[2]\pi_n^{coll}))} \tag{5}$$

By setting λ, we are able to choose the input range of the exponential function. In order to get a probability in the range of $[e^{-\lambda}, 1]$, all weights $\omega[i]$ are determined experimentally and normalized to a sum of 1. Also, all π_n are normalized to have a sum of 1. Then, we are able to compute a weighted mean of all particles P to estimate the current hand configuration s_{mean}.

$$s_{mean} = \sum_{n=0}^{N} \pi_n \cdot s_n \tag{6}$$

E. Partitioning Search Space in Resampling

For the next iteration of the particle filter, we have to resample our set of particles. Basically, we use factored resampling of the old particle set, as described by Isard [10]. Using this approach, we could realize hand-tracking, but not tracking of individual fingers. We encountered the problem that a particle whose hand orientation configuration is misaligned will degrade the rating of the finger tracking, because it might compare a finger in the model to a neighboring finger in the image.

To improve performance, we propose to divide our particle set into two disjunct sets. The first set is used to track the posture of the hand given the current finger configuration, so we assign the mean finger configuration from the last iteration to this particle set. Posture is resampled from all particles considering their weight. The second set is used to track the fingers at the last known hand posture. For this, the mean hand posture of the last iteration is assigned to all particles of this set. Finger configuration is then resampled from all particles. After resampling, both sets are merged.

Using this approach, we get the benefit of concentrating particles in the last known position to search for finger configuration in the second set, thereby practically reducing dimensionality of the search space by six degrees of freedom. At the same time, we search for the new hand posture using the last known finger configuration using the first set. Assigning the means to the particles was done this way to preserve the particle filter's property to express multiple hypothesis.

F. Distributed Real-time Tracking

Particle Filters are well suited for parallelisation on clusters, because the computationally expensive rating of each particle is independent from all other particles. Nevertheless, communication is necessary to spread the particles onto all cluster nodes, and to gather the corresponding probabilities. In our case, we also have to distribute the preprocessed images.

Our cluster nodes are interconnected via standard Gigabit ethernet. We used broadcasts to minimize the necessary traffic

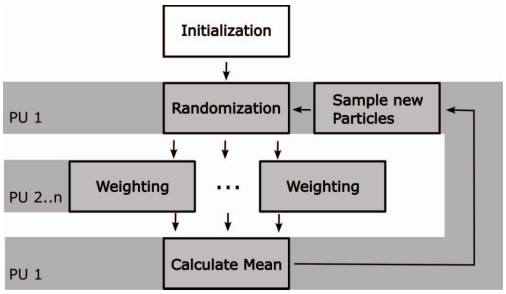


Fig. 5. Structure of the distributed Particle Filter. Each Pu stands for a separate cluster node.

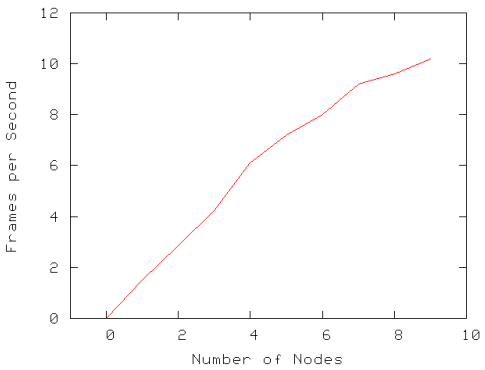


Fig. 6. Scaling of our tracking system.

for sending images. To implement the distributed particle filter, we use the Distributed Vision Cluster proposed by [15]. By starting multiple rating threads on each node, we can also profit from multi processor and multi core systems. We found that interleaving the rating of stereo images to only one image for each iteration of the particle filter almost doubles performance. For all our tests and benchmarks, we used an empirically determined number of 3000 particles. Using more particles did not improve tracking considerably, and using less than 2000 particles considerably degraded the result. The Distributed Vision Cluster [15] enabled the particle filter to scale with a speedup of 6.84 using 9 Cluster Nodes, that is an efficiency of 76.1%. We were able to reach 10.2 FPS. For our measurements, our Clusternodes used dual 2.16 Ghz CPUs. A performance measure can be seen in Figure 6.

V. Results

Evaluating the tracking performance of complex articulate objects is a challenging task, due to the facts that usually one can not easily obtain a ground truth of video sequences. For hand tracking, one also cannot easily use a more accurate method like a data glove or markers, as it changes the appearance of the hand. In our studies we have elected to conduct our evaluations in twofold: First, we use natural video to prove that our tracker works with real-world images by tracking the transition between postures. To show the tracker's performance, we generated artificial video sequences with known ground truth to evaluate the tracking error of the system.

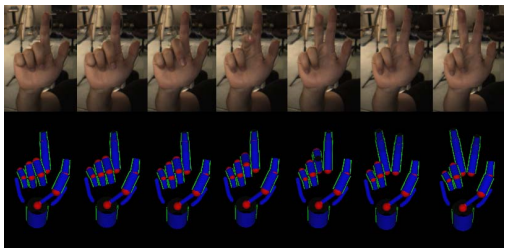


Fig. 7. Tracked Sequence

	1	2	3	4	5
1	-	2	0	7	3
2	6	-	0	12	3
3	6	5	-	0	7
4	1	9	1	-	4
5	9	0	5	13	-

TABLE I
FRAMES REQUIRED FOR CONVERGENCE

A. Natural Video Sequence

We selected 5 hand postures, and tracked the transition between each pair. We noted the number of frames necessary until the tracker converged completely to the correct posture, after the transition ended. For an example transition see Figure 7. The capture rate of all videos was 30 fps.

In Table I, we outlined the required number of frames from posture[column] to posture[row] transitions. The average number of required frames to converge is 4.65. The converged finger configuration was always correct, even with a slight palm offset.

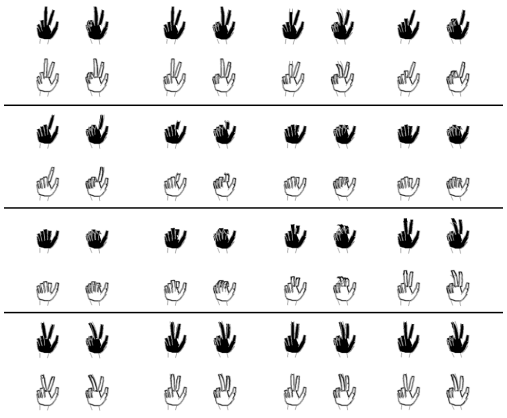


Fig. 8. Tracking sequence for the synthetic hand stereo projection (frame 150 to 400). First row shows the input for the area cue, the second row the input for the edge cue.

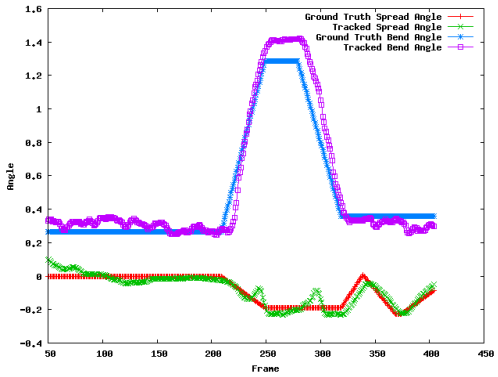


Fig. 9. Comparison of estimated values and ground truth for the two DOF of the index finger's MCP.

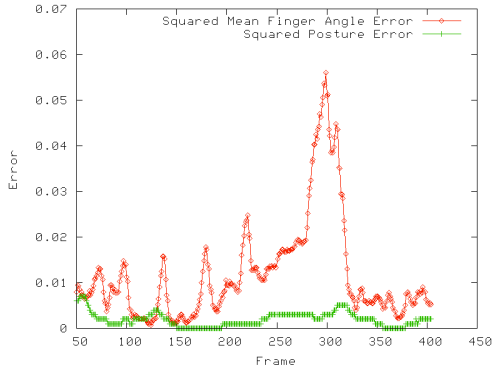


Fig. 10. Error graph showing the mean squared error for θ_{MCP}, and for the posture of the hand (tracked angles are in radians).

B. Synthetic Video Sequence

In our second experiment, we use synthetic videos, that have been created by projecting a stereo animation sequence of the 3D hand model (Figure 8). From the sequences presented, we can derive the ground truth of each sequence, therefore, a comparison of the tracked kinematic parameters with ground truth are made. Figure 9 shows that the index finger is tracked over 400 frames. In Figure 10, the posture error of the hand can be seen for this sequence. The squared mean error of all fingers for tracking θ_{MCP} reached only as highest 0.06 at frame 300.

VI. Conclusion

In this paper, we presented a markerless hand and finger tracking system using particle filters. We proposed a new z-buffer based method to correctly rate cues of articulate self-occluding objects like a human hand. To be able to track fingers accurately, we improved performance by proposing a particle resampling method to partition the search space. Implementation was conducted using a distributed particle filter on a PC-cluster, enabling us to achieve tracking results at high frame-rates. The effectiveness of the tracking system is demonstrated with the real-time control of a 20 degrees of freedom dextrous robot hand.

The current tracking system could greatly benefit from the combination of a full-body human tracker [17]. The arm posture could be utilised to obtain the position of the wrist, thus greatly reduce the search space for hand orientation. This forms part of our future work.

Acknowledgment

The work described in this paper was partially conducted within the EU Cognitive Systems project PACO-PLUS (FP6-2004-IST-4-027657) funded by the European Commission.

References

[1] D. J. Sturman and D. Zeltzer, "A survey of glove-based input," *IEEE Computer Graphics and Applications*, vol. 14, no. 1, pp. 30–39, 1994.

[2] P. Dreuw, "Appearance-based gesture recognition," Diploma Thesis, RWTH Aachen University, Aachen, Germany, January 2005.

[3] Y. Wu and T. S. Huang, "Vision-based gesture recognition: A review," *Lecture Notes in Computer Science*, vol. 1739, p. 103, 1999.

[4] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "A review on vision-based full dof hand motion estimation," in *Computer Vision and Pattern Recognition (CVPR)*, 2005, p. 75.

[5] B. Stenger, A. Thayananthan, P. Torr, and R. Cipolla, "Hand pose estimation using hierarchical detection," *Lecture Notes in Computer Science International Workshop on Human-Computer Interaction*, vol. 3058, pp. 102–112, 2004. [Online]. Available: citeseer.ist.psu.edu/stenger04hand.html

[6] J. Rehg and T. Kanade, "Digiteyes: Vision-based hand tracking for human-computer interaction," in *Workshop on Motion of Non-Rigid and Articulated Bodies*, 1994, pp. 16–24. [Online]. Available: citeseer.ist.psu.edu/rehg94digiteyes.html

[7] H. Ouhaddi and P. Horain, "Hand tracking by 3d model registration," in *International Scientific Workshop on Virtual Reality and Prototyping*, Laval, France, 1999, pp. 51 – 59. [Online]. Available: citeseer.ist.psu.edu/ouhaddi99hand.html

[8] Y. Wu and T. S. Huang, "Capturing articulated human hand motion: A divide-and-conquer approach," in *International Conference Computer Vision (ICCV)*, vol. 1, 1999, p. 606.

[9] B. Stenger, P. R. S. Mendonca, and R. Cipolla, "Model-based 3d tracking of an articulated hand," in *Computer Vision and Pattern Recognition (CVPR)*, 2001, p. 310.

[10] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998. [Online]. Available: citeseer.ist.psu.edu/isard98condensation.html

[11] J. Lin, Y. Wu, and T. S. Huang, "Capturing human hand motion in image sequences," in *Workshop on Motion and Video Computing*, vol. 00. IEEE Computer Society, 2002, p. 99.

[12] J. Lee and T. L. Kunii, "Model-based analysis of hand posture," *Computer Graphics and Applications*, vol. 15, no. 5, pp. 77–86, 1995.

[13] N. Shimada, K. Kimura, and Y. Shirai, "Real-time 3-d hand posture estimation based on 2-d appearance retrieval using monocular camera," *ratfg-rts*, vol. 00, p. 0023, 2001.

[14] A. Ude, C. Gaskett, and G. Cheng, "Foveated vision systems with two cameras per eye," in *International Conference on Robotics and Automation (ICRA)*, May 2006.

[15] A. Ude, V. Wyart, M. Lin, and G. Cheng, "Distributed visual attention on a humanoid robot," in *International Conference on Humanoid Robots (Humanoids)*, 2005.

[16] J. Deutscher, A. Blake, and I. Reid, "Articulated Body Motion Capture by Annealed Particle Filtering," in *Computer Vision and Pattern Recognition (CVPR)*, Hilton Head, USA, 2000, pp. 2126–2133.

[17] P. Azad, A. Ude, T. Asfour, G. Cheng, and R. Dillmann, "Image-based Markerless 3D Human Motion Capture using Multiple Cues," in *International Workshop on Vision Based Human-Robot Interaction*, Palermo, Italy, 2006.